Differential Privacy in Telco Big Data Platform

Xueyang Hu[†,⋆], Mingxuan Yuan[⋆,+], Jianguo Yao[†,+], Yu Deng[†], Lei Chen[‡], Qiang Yang[‡], Haibing Guan[†] and Jia Zeng[◦,⋆,+]

[⋆]Huawei Noah's Ark Lab, Hong Kong
[†]Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University
[‡]Department of Computer Science and Engineering, The Hong Kong University of Science and Technology
[◦]Collaborative Innovation Center of Novel Software Technology and Industrialization, Soochow University

[+]Corresponding Authors: yuan.mingxuan@huawei.com, jianguo.yao@sjtu.edu.cn, zeng.jia@acm.org

ABSTRACT

Differential privacy (DP) has been widely explored in academia recently but less so in industry possibly due to its strong privacy guarantee. This paper makes the first attempt to implement three basic DP architectures in the deployed telecommunication (telco) big data platform for data mining applications. We find that all DP architectures have less than 5% loss of prediction accuracy when the weak privacy guarantee is adopted (e.g., privacy budget parameter $\epsilon \geq 3$). However, when the strong privacy guarantee is assumed (e.g., privacy budget parameter $\epsilon \leq 0.1$), all DP architectures lead to $15\% \sim 30\%$ accuracy loss, which implies that real-word industrial data mining systems cannot work well under such a strong privacy guarantee recommended by previous research works. Among the three basic DP architectures, the Hybridized DM (Data Mining) and DB (Database) architecture performs the best because of its complicated privacy protection design for the specific data mining algorithm. Through extensive experiments on big data, we also observe that the accuracy loss increases by increasing the variety of features, but decreases by increasing the volume of training data. Therefore, to make DP practically usable in large-scale industrial systems, our observations suggest that we may explore three possible research directions in future: (1) Relaxing the privacy guarantee (e.g., increasing privacy budget ϵ) and studying its effectiveness on specific industrial applications; (2) Designing specific privacy scheme for specific data mining algorithms; and (3) Using large volume of data but with low variety for training the classification models.

1. INTRODUCTION

Telecommunication (telco) big data record billions of customers' communication behaviors for years in the world. Mining big data to increase customers' experience for higher profits becomes one of important tasks for telco operators (e.g., telco churn prediction with big data [12]). To this end, telco operators aim to build big data platforms to analyze patterns of customers' life-cycle behaviors for the next-generation business intelligence. For customers, most telco data are privacy-sensitive such as call detailed records (CDRs), billing records, purchase history, payment records, mobile search queries, social networks, and trajectory information [14]. Therefore, individual customer's privacy protection module is a necessary component in telco big data platform. Differential privacy (DP) [6] is the state-of-the-art privacy protection technique in academia, because it provides mathematically a very strong privacy protection guarantee, which ensures the outcome of any authenticated query/calculation to be insensitive to any individual record in the database (DB). However, very few successful DP cases have been reported in real-world large-scale industrial data mining (DM) projects possibly because DP's strong privacy guarantee often causes worse DM performance.

In this paper, we make the first attempt to implement three basic DP architectures in the deployed telco big data platform for churn prediction [12]. Customer churn is perhaps the biggest challenge in telco industry. A churner quits the service provided by operators and yields no profit thereafter. The churn prediction system provides a list of customers (ranking by the churn likelihood) who will most likely churn in the next month, which requires feature engineering and classifier learning on customers' historical communication records. In industry, data-driven churn prediction generally includes constructing useful features (aka predictor variables) and building good classifiers (aka predictors) or classifier ensembles with these features [9, 36]. We would like to use churn prediction over telco data as an example to study DP mechanisms. This would help to answer how DP will perform with real industry big data mining if they outsource mining tasks to third parties. For privacy protection, we analyze and evaluate three DP implementations for decision trees (DTs) in the churn prediction system with big data. The reason we choose DTs is that they have been widely used in industrial data mining systems with good prediction performance [10, 30, 9, 36, 28]. For example, DTs have been used for churn analysis [32], domain knowledge integration [19], voltage reduction [1], and so on. Another widely used classifier ensemble method, random forest (RF) [2], is composed of a group of DTs with the similar privacy protection strategies. Although RF often achieves a better prediction performance [16, 26, 12], it is still necessary to analyze the effectiveness of DP techniques for DTs.

To summarize, we make the following contributions on DP from industrial perspectives:

- Implementation in telco big data platform: We broadly categorize the recent DP implementations for DTs into three basic architectures: 1) Data Publication Architecture [25, 39]; 2) Separated Architecture [7]; and 3) Hybridized Architecture [7, 13]. We will describe detailed implementations of the three basic DP architectures in Section 4.

Proceedings of the VLDB Endowment, Vol. 8, No. 12

Table 1: Notations & Meanings.

Notation	Meaning	Notation	Meaning
DB	Database	DT	Decision Tree
DM	Data Mining	RF	Random Forest
DP	Data Privacy	AUC	Area under ROC Curve

- Extensive experimental results on big data: 1) We study the influence of privacy budget parameter on different DP implementations with industrial big data. The accuracy trade-off testing with different privacy budgets is similar to the previous works [7]; 2) We compare the performance of three basic DP architectures in churn prediction; 3) We examine how volume and variety of big data affect the performance of DP, where volume and variety are two basic characteristics of big data; and 4) We compare the DP implementation performance between the simple DT and the relatively complicated RF in churn prediction. More details can be found in Section 5.

- Important observations in churn prediction: 1) All DP architectures have a relative accuracy loss less than 5% with week privacy guarantee (e.g., privacy budget parameter $\epsilon \geq 3$). However, when the privacy guarantee becomes stronger (e.g., $\epsilon \leq 0.1$), the relative accuracy loss is as large as $15\% \sim 30\%$; 2) Among all three basic DP architectures, the Hybridized Architecture performs the best because of its specific privacy design for a specific data mining algorithm such as DTs; and 3) The prediction error caused by the DP protection increases with the growth of the number of used features, but decreases with the growth of the training data volume (the number of instances used to train the model).

- Practical suggestions on deployment of DP in large-scale industrial data mining systems: 1) Relaxing privacy guarantee (e.g., increasing privacy budget parameter ϵ) and study its effectiveness on specific industrial applications; (2) Designing specific privacy scheme for a certain data mining algorithm; and (3) Using large volume of data but with low variety for model training.

Table 1 summarizes the notations used in this paper. The rest paper is organized as follows. Section 2 reviews the related work on privacy protection techniques. Section 3 introduces the real-world telco big data platform and the customer churn prediction component. Section 4 describes industrial DP implementations for decision tree (DT) based data mining algorithms. Section 5 reports extensive experimental results on different DP techniques using different parameter settings. Section 6 draws conclusions and discusses possible research directions in future.

2. RELATED WORK

Anonymization [3, 4] is the first-generation privacy protection technique, which removes or replaces the explicitly sensitive identifiers (ID) of customers, such as the identification number or mobile phone number, by random mapping or encryption mechanisms in DB, and provides the sanitized dataset without any ID information to DM services. However, anonymization still discloses individual customer's privacy. For example, Sweeny [29] demonstrated that an individual's name in a public voter list can be linked with his/her record in a published anonymized medical record through the combination of some attributes including zip code, birthday and sex. The attributes that can be used to identify a user or a small group of users is called the *quasi-identifier*. Sweeney [29] found that 87% of the U.S. population has disclosed information that may be uniquely distinguished by their quasi-identifiers.

To avoid the attack using quasi-identifiers, K-Anonymity [29, 5, 15] is invented to provide stronger privacy protection, which ensures that any quasi-identifier in the published dataset appears at least K times. K-Anonymity guarantees whenever an attacker uses the quasi-identifier to attack a user, he/she will always obtain at least K similar candidates. To make any quasi-identifiers appear at least K times, K-Anonymity generates, permutes or changes the quasi-identifier values. However, K-Anonymity still has some weaknesses. For example, if the sensitive information is the disease and the group of customers with the same quasi-identifiers have the same disease HIV, the privacy information is still disclosed. So, the L-diversity models [21, 22] are proposed, which require the sensitive information in an anonymous group (aka a group of customers with the same quasi-identifiers) must have enough "diversity". Similarly, stronger privacy protection models, such as T-closeness [17], are proposed (the distribution of sensitive information distribution in any anonymous group must be close enough to the distribution of the whole dataset). Researchers also develop different privacy protection models for other scenarios, such as the graph dataset [20, 37] (the linkage and node label information can be treated as both sensitive and quasi-identifier) and continuously published data [33]. These solutions can be regarded as the second-generation privacy protection techniques, which publish a sanitized dataset with certain anonymity or diversity requirements. The biggest weakness of the second-generation protection techniques is that they must predefine the background knowledge (quasi-identifier) of the attacker (the knowledge that an attacker will use to attack an individual customer's privacy). If the background of attackers is unknown, the protection may totally fail and the data provider cannot control anything after the sanitized dataset has been published. In real-world applications, it is difficult to define the quasi-identifiers and sensitive information.

Unlike previous solutions, DP [6] is currently the strongest privacy protection technique, which does not need any background information assumption of attackers. The attacker can be assumed to know the maximum knowledge, e.g., he/she knows all the other instances in the DB except the targeting one. DP ensures the outcomes of the authenticated queries/calculations to be insensitive to any individual record in the DB. Insensitivity means when the attacker observes the output of DB, the probability he/she learns an individual customer is in the DB and the probability he/she learns this individual user is not in the DB should be indistinguishable. In this situation, each customer is provided the strongest privacy protection. For this reason, DP can be viewed as the third-generation privacy protection technique. The research community prefers DP due to its strong mathematical boundary of the leaked privacy [6, 23, 35, 27, 11, 18, 38, 34]. Recently, DP has been studied in different scenarios including histogram query [35], statistical geospatial data query [27, 11], frequent item set mining [18, 38] and crowdsourcing [11]. How to adapt DP on decision trees are studied by [7, 13]. Xiao [34] designed a new graph data publication model, in which each graph is described as a tree structure. DP is used to protect the tree structure as well as the probability value at each tree node. Erlingsson [31] studied how to use DP for crowdsourcing statistics from end-user client software. Machanavajjhala [23] investigated the relationship between privacy and accuracy of personalized social recommendations, where DP is considered as the privacy measurement. Interestingly, for majority of nodes in the network, recommendations must either be inaccurate or violate DP assumptions.

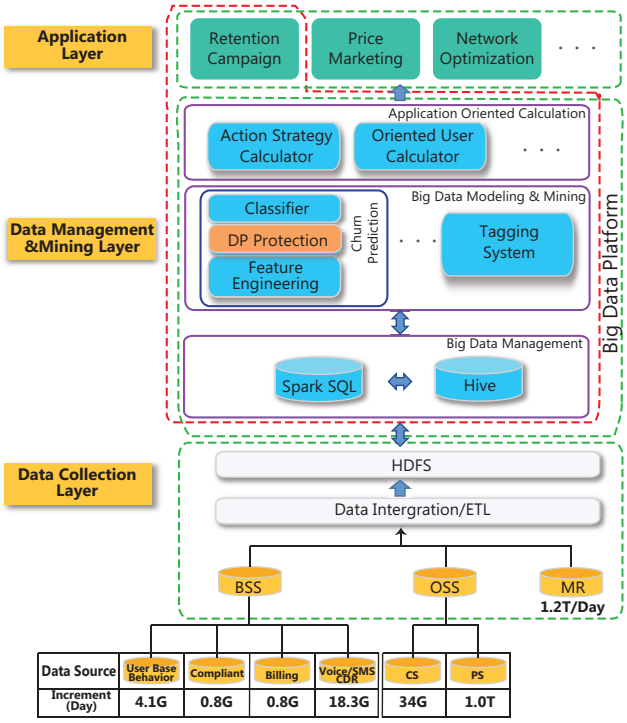


Figure 1: The overview of telco big data platform.

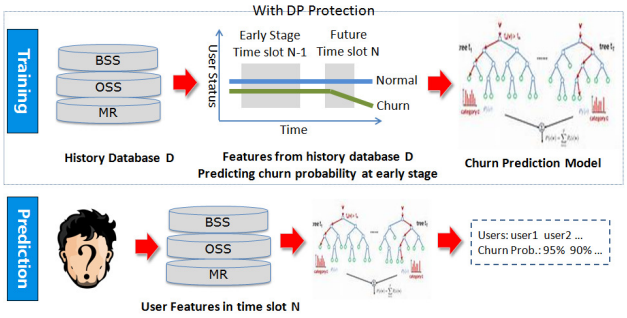


Figure 2: Classifier training and prediction.

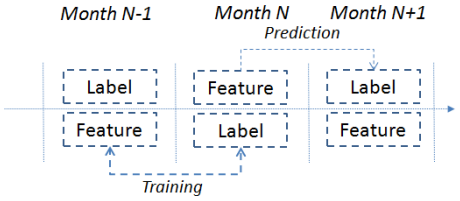


Figure 3: The sliding window setting for churn prediction.

To the best of our knowledge, there are few DP techniques implemented in deployed industrial DM services. Since DP has shown its benefits by providing the strongest privacy protection without predefining the attackers' background knowledge, it is worthwhile investigating its performance in real-world industrial systems. Telco big data platform analyzes large-scale customers' communication behaviors with privacy-sensitive information (e.g., billing, social networks and trajectories), which motivates DP implementations with a strong privacy guarantee for important applications such as churn prediction in this paper. In the future, we may explore and compare first-, second- and third-generation of privacy protection techniques in the telco big data platform.

3. TELCO BIG DATA PLATFORM

We implement three basic DP architectures in telco big data platform for churn prediction. Figure 1 illustrates the platform deployed in one of biggest telco operators in China [12]. The structure of platform is composed of three layers: *Data Collection Layer*, *Data Management & Mining Layer* and *Application Layer*. The Data collection layer is used for gathering all types of telco data. Data Management & Mining Layer is used for data management including information integration, feature extraction and classifiers to support various business needs in the application layer. For example, the retention campaign component in the application layer is supported by churn predictor in data management & mining layer. Other business needs include price marketing and communication network optimization.

In general, telco big data come from three types of resources, i.e., *business supporting system* (BSS), *operation supporting system* (OSS), and *measurement report* (MR). BSS has four basic functions: product management, order management, revenue management and customer management. BSS data include billing, short message service record, call records, complaint records recharge history, and customers' demographic data. OSS manages communication network functions including network inventory, service provisioning, network configuration and fault management. OSS data include two categories, circuit switch (CS) and packet switch (PS). CS is related to the voice service supporting system. CS data reflect the voice service quality. PS is related to the mobile internet data service supporting system. PS data is also called mobile broadband (MBB) data, which are gathered by probes with deep packet inspection (DPI) technique. PS data describe customers' data usage behaviors such as mobile search queries, app usages, and streaming records. They also reflect the data service quality such as web speed and connection success rate. MR data contain the records of signal strengths and angles from a mobile device to its nearby six cell towers (The phone device will decide to choose which cell tower to connect based on the measurement). They are collected from radio network controller (RNC) and can be used to estimate customers' approximate positions of the phone for trajectories [14]. The data volume gathered per day in the platform is also shown in Figure 1, where BSS data are around 24 GB, and OSS/MR data are around 2.2TB per day. Both OSS and MR data occupy over 90% data volume of the entire telco big data. After integrating the data by extraction-transformation-loading (ETL) tools, we store the raw data in Hadoop distributed file system (HDFS). We see that BSS, OSS and MR data are privacy-sensitive to each customer, so that privacy protection techniques are required in telco big data platform for different applications. Moreover, privacy protection becomes one of the major concerns for telco operators to monetize their data assets, which has not been fully discussed and considered in telco big data platform in previous work [12].

In Figure 1, we focus on DP implementations for churn prediction component, which is a classifier to predict customer's churn probabilities in the next month. The retention campaign is automatically issued on the targeted customers with high probabilities every month. Figure 2 shows the flow chart of classifier training and prediction phases. First, we extract relevant feature vectors of customers from BSS, OSS and MR data. Second, we assign class labels $l = \{0, 1\}$ ($l = 1$ for churners) to all customers according to

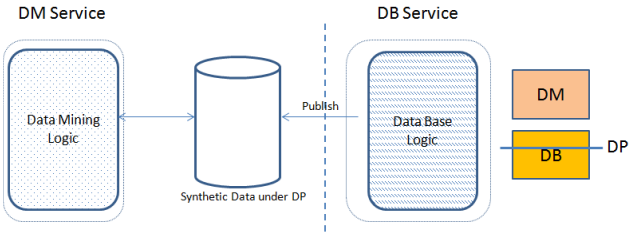
some business rules. For example, we assume prepaid customers as churners if they do not recharge within 15 days after their balance is below zero. Finally, we train a DT classifier or RF classifier based on labeled features. In the prediction phase, we input the customers' feature vectors (not for training) to the classifier and output the churn probabilities. Figure 3 shows how the deployed churn predictor works in the sliding window settings. First, we use month N to label features (churner or non-churner) in month $N-1$. Second, we use the labeled features in month $N-1$ to train a classifier. Finally, we extract the features in month N and input them into the classifier to predict the label (potential churners) in month $N+1$. The churn prediction system works well on original customers' data without privacy protection. For example, the precision reaches 0.96 for the top $50K$ potential churners with highest probabilities. More details can be found in [12]. However, as we discussed before, it is necessary to consider privacy protection in DM because all customers' data are privacy-sensitive. It should be emphasized that under our DP schemes, the privacy of people in the training data is protected, but the privacy of people in the prediction data (that is, the data which you will apply the trained model to) is not. In a DM task, data engineers are involved in the feature and model design (model training). A user's data, which are used to train a model, do not bring any benefit to the user at this stage. Thus the privacy of people in the training data must be protected. In the prediction stage, the service is issued based on each individual user. The features of each individual should be extracted and exported into the model. The DP does not fit with this scenario. However, the privacy protection requirement is not as important in prediction as in model training. One reason is that each user could get some benefits from the prediction/recommendation in many scenarios. Thus a service contract can be made to each user to get a service by providing personal data. The other reason is that after the model is trained, the data mining engineers do not need to be involved in the prediction stage. The model can be deployed on a security guaranteed platform and automatically runs. In this case, privacy protection is not necessary. In view of above mentioned reasons, we think for prediction/recommendation systems, DP protection should only be considered in training stage.

4. INDUSTRIAL DP IMPLEMENTATIONS

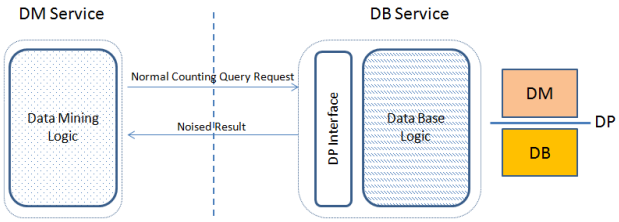
DP [6] is a privacy definition which ensures the outcome of any authenticated query/calculation to be insensitive to any individual record in the DB. DP is preferred due to its strong mathematical boundary of the leaked privacy and has been widely studied in the research community [6, 23, 35, 27, 11, 18, 38, 34]. The implementations of DP can be broadly categorized into three basic architectures: (1) the Data Publication Architecture; (2) the Separated (DM and DB) Architecture; and (3) the Hybridized (DM and DB) Architecture. We illustrate the concepts of these three basic architectures in Figure 4.[1] We see that the major difference of three basic architectures lies in the position of DP interface between DM and DB on the right panel of Figure 4.

In the Data Publication Architecture, the DB service uses a specific schema to publish a synthetic dataset with the DP guarantee from the real original dataset. In this way, the DP interface is implemented within the DB service between original and synthetic datasets in Figure 4(a). Since the synthetic dataset is privacy-insensitive, any DM service can be directly applied on the top of the published and protected synthetic dataset. The benefit of this

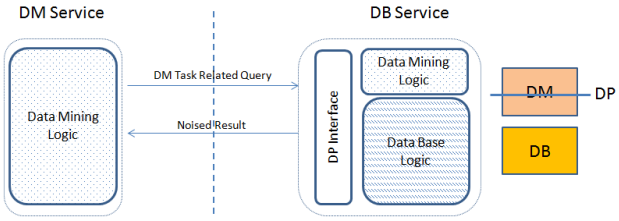
[1]In Figure 4, we illustrate the concepts of three DP architectures. We will show some detailed implementations of DP for decision trees in Section 4.3.



(a) Architecture 1: Data Publication Architecture



(b) Architecture 2: Separated (DM and DB) Architecture



(c) Architecture 3: Hybridized (DM and DB) Architecture

Figure 4: Three basic DP architectures.

architecture is that all DM algorithms can be used without privacy concerns. However, the shortcoming is also quite clear: the DM service runs on the synthetic dataset instead of the real/original one, so that the mining quality is seriously restricted by the schema of generating synthetic dataset under the DP guarantee.

The Separated Architecture is an implementation that separates the DB service from the DM service through a DP interface in Figure 4(b). The DB provides the query interface, which supports the traditional aggregation queries (more accurately, the counting queries) with the DP guarantee. The DB service has no idea about how DM service will use the results of these queries. The benefit of this system is that the traditional DB structure does not need any change to support specific DM services. Since the DM services are specifically designed to use these query results, the system accuracy is expected to be higher than the Data Publication Architecture. However, since the DM services are on the top of aggregation queries, they cannot be implemented and optimized beyond the scope of traditional queries. This may cause some design limitations of the DM services and lead to some accuracy loss.

The Hybridized architecture adapts only the DP interface into DM services. In this situation, the DB service is designed to support some specific queries (such as the best splitting point selection query in DTs) for specific DM services in Figure 4(c). The benefit of this architecture is the DP implementation is optimized for a specific DM method. So, the accuracy of the DM is expected to be the highest among the three basic architectures. The shortcoming is that the logics of both DM and DB services depend closely. The DB developers must handle extra types of queries for specific DM services, which are different from the traditional ones supported by DB services.

4.1 Differential Privacy

Here we provide a brief introduction to DP's basic concept, characteristic and the two applying mechanisms, which will guide our implementations of DP in the churn prediction system. DP is mathematically defined as follows. When the attacker observes the output of DB, the probability he/she learns an individual customer is in the DB and the probability he/she learns this individual customer is not in the DB should be indistinguishable, which is formally defined as

DEFINITION 1. *A randomized function/query/calculation f provides $\epsilon-$differential privacy if for any neighboring data bases D_1 ($D_1 \triangle D_2 = 1$) and D_2, for any output $O \in Range(f)$, $Pr[f(D_1) \in O \leq e^{\epsilon} \times Pr[f(D_2) \in O]]$.*

Neighboring DBs D_1 and D_2 are two DBs, where there are only one individual record difference between them ($D_1 \triangle D_2 = 1$). The parameter ϵ is the privacy budget, which can be used to control the level of privacy protection. The smaller the value of ϵ is, the stronger privacy protection it provides. DP guarantees that the query result is insensitive to any individual record. The probability that an attacker guess an individual record is in or not in the database in at most e^{ϵ} from the outputs of queries/calculations. The DP satisfies a *composability* property [24] defined as follows,

THEOREM 1. *Composability Property: Let f_i each provide ϵ_i- differential privacy privacy. The sequence of $f_i(D)$ provides $(\sum_i \epsilon_i)$- differential privacy.*

Therefore, the ϵ parameter can be considered as an accumulative privacy cost as more queries are executed [8]. These costs keep accumulating until they reach an allotted privacy budget [8].

There are two mechanisms to realize DP, the *Laplace mechanism* [6] and the *Exponential mechanism* [24]. Both of them need to calculate the global sensitivity of a function f. The global sensitivity of a real-valued function is used to represent the maximum possible change of its output value when adding or removing a single individual record.

DEFINITION 2. *The global sensitivity of a function $f : D \rightarrow \Re^d$ is $\triangle f = \max_{D_1, D_2 \text{with} D_1 \triangle D_2 = 1} \parallel f(D_1) - f(D_2) \parallel_1$.*

The Laplace mechanism is used to realize DP by adding noise to the outcome of the queries which return real values. The noise is drawn from a Laplace distribution with the probability $pr(x|\lambda) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}$, where $\lambda = \frac{\triangle f}{\epsilon}$.

THEOREM 2. *Laplace Mechanism: Given a function $f : D \rightarrow \Re^d$, the computation M, $M(D) = f(D) + (Laplace(\frac{\triangle f}{\lambda}))^d$ provides $\epsilon-$differential privacy.*

For example, a function (or a query) is defined as counting the number of records in database D. Obviously the global sensitivity of this function is 1. A disturbed result, $|D| + Laplace(\frac{1}{\epsilon})$, is returned. This result guarantees the $\epsilon-$differential privacy of this counting function.

Besides real-valued functions, there are functions exporting non-real values. In this scenario, *Exponential mechanism* is used to provide $\epsilon-$differential privacy. Exponential mechanism samples an o from the output space O according to a quality function q that scores outcomes of a functions, where higher scores are better. The quality function gives a probability distribution over the output domain, it samples the outcome according to this distribution to close to the optimum output while ensuring the $\epsilon-$differential privacy.

THEOREM 3. *Exponential Mechanism: Given a quality function $q : (D \times O) \rightarrow \Re$, which assigns a score to each outcome $o \in O$, let $\triangle q = \max_{o, D_1 \triangle D_2 = 1} \parallel q(D_1, o) - q(D_2, o) \parallel$, the computation M, M(D,q)=$\{$return o with probability $\propto \exp(\frac{\epsilon q(D,o)}{2\triangle q})\}$ provides $\epsilon-$differential privacy.*

Table 2: An example of training instances.

Gender	ARPU	3G	Churn
F	low	yes	yes
M	low	no	yes
M	high	no	yes
M	high	no	yes
F	high	yes	no
F	high	yes	no
F	high	yes	no
M	high	yes	no

4.2 Decision Trees

Since we focus on DT based models, in this section, we give a brief introduction to them to identify the key calculation related with DP in the DT construction algorithm. In the next section, we'll introduce how to implement DT models with the three different architectures.

Decision trees (DTs) are a category of widely used classifiers [10, 30, 9, 36, 28]. The basic idea is to use one or a group of DTs to map observations about an instance (features of an instance, aka the attributes to describe an instance) to predict about the instance's target label (e.g., whether a customer will be a churner). A DT is a mechanism to organize the observed instances (training instances to build a DT) in a tree structure space. Each instance $I_i = (l_i, F_i = [f_{i,1}, f_{i,2}, ..., f_{i,m}])$ is composed of a label and a group of features (attributes). For example, in Table 2, each row denotes an instance for one record of a customer. The first instance is (Churn=*yes*,[Gender=*F*, ARPU=*low*, 3G=*yes*]), where $Churn$ denotes the class label, $ARPU$ is the average revenue per unit (customer) and $3G$ means the 3G service. The feature space of the dataset can be organized as a decision tree structure as shown in Figure 5, When a new instance (customer) appears, such as a customer with features [Gender=*M*, ARPU=*low*, 3G=*yes*], the DT can be used to predict the class label of this customer by searching the tree. There may exist exponential (according to the number of features) number of trees by selecting different attributes in branches. It is intuitive that the purer a leaf node is (aka most of instances in this leaf node have the same class label), the more confidential of the prediction can be made. Hence, a good DT always tends to cut the feature space as pure as possible in the leaf nodes. When generating a DT, we always prefer to select a feature, which can generate the purest subspace by splitting the feature space (aka creating new branches in a tree node).

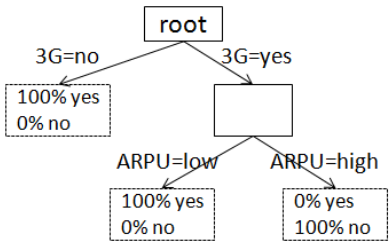


Figure 5: An example of a DT.

The algorithm skeleton of a DT generation is shown in Algorithm 1 [30], which works recursively by selecting the best feature

Algorithm 1: The DT generation algorithm skeleton.

Input: A dataset with each item $I_i = (l_i, F_i = [f_{i,1}, f_{i,2}, ..., f_{i,m}])$
Output: A decision tree T, which can be used to predict the label l_j of a new instance I_j given a new instence $F_i = [f_{j,1}, f_{j,2}, ..., f_{j,m}]$.

```
if stopping_cond(D) = true then
    leaf = createNode();
    leaf.label = Classify(D);
    return leaf;
else
    root = createNode();
    root.test_cond = find_best_split(D);
    let V={v|v is a possible outcome of root.test_cond};
    for each v ∈ V do
        D_v={e|root.test_cond(e)=v and e ∈ D};
        child=TreeGrowth(D_v);
        add child as decendent of root and label the edge (root → child) as v;
```

to split the training records until the stopping condition of generating a leaf node is achieved [30]. In the algorithm [30], **createNode()** is used to extend the DT by creating new nodes/branches. How to create new branches are controlled by **find_best_split()**. It determines which feature should be selected for splitting the training record. The choice depends on the impurity measurement used to determine the goodness of a split. Some widely used measures include entropy and the Gini index etc. The **Classify()** function is used to determine the label assigned to a leaf node. For each leaf node t, let $p(l|t)$ denote the fraction of training instances from label i associated with the node t, the label l with the maximum $p(l|t)$ can be assigned to node t or $p(l|t)$ is directly assigned as the probability that node t is likely to have label l. The **stopping_cond()** function is designed to terminate the tree construction by testing whether all the instances have the same label or the number of instances has fallen below some minimum threshold.

There are a variety of implementations of the DT generation algorithm according to different impurity measurements. Generally, there are three widely used impurity measurements:

- **Information Gain**: the ID3 algorithm [10] uses the change of entropy before and after the data is split on feature A. The entropy $H(D)$ is a measurement of the uncertainty in the dataset D, i.e., $H(D) = -\sum_{l \in L} p(l) \log_2 p(l)$, where l denotes the class label, and $p(l)$ is the proportion of the number of instances that have label l in D. The information gain after splitting the dataset D by feature A is $IG(D, A) = H(D) - \sum_{a \in A} p(D_a) H(D_a)$. In this formula, a is a value of feature A, which will form a new branch and D_a is the new dataset that follows this branch. $p(D_a)$ is the proportion of the number of instances in D_a to the number of instances in D ($\frac{|D_a|}{|D|}$).

- **Information Gain Ratio:** One problem of the ID3 algorithm is that it tends to split the data by the feature which has a lot of values (e.g. the ID number). To avoid this, the C4.5 algorithm [10] uses the information gain ratio to measure the impurity. Information gain ratio is calculated as $GainRatio(D, A) = \frac{IG(D,A)}{SplitInfo(S,A)}$. *SplitInfo* is defined as $SplitInfo(D, A) = -\sum_{a \in A} \frac{|D_a|}{|D|} \log_2 \frac{|D_a|}{|D|}$. The C4.5 algorithm handles both continuous and discrete attributes.

- **Gini Index:** The CART algorithm [10] uses the Gini Index. Given a dataset D, the Gini index of D is defined as $Gini(D) = 1 - \sum_{l \in L} p(l)^2$. We can use $Gini(D, A) = \sum_{a \in A} \frac{|D_a|}{|D|} Gini(D_a)$ to represent the Gini index of dividing the dataset by feature A. The change of Gini index is calculated as $\triangle Gini(D, A) = Gini(D) - Gini(D, A)$. The decision will select the feature with the largest $\triangle Gini$ to create new branches.

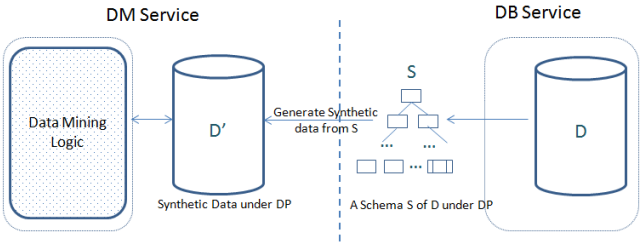


Figure 6: Data Publication Architecture for DTs.

The key calculation (the step that needs to interact with the DB) in DT generation is to find the best branch creation feature according to impurity measurements. In the next section, we introduce how different DP techniques implement this calculation. The complicated classifier ensembles such as RF [2] are composed of a group of DTs. RF generates a DT on bootstrap samples of the original dataset D on a sampled subset of features and aggregates the prediction result of all DTs. Bootstrap means random sampling with replacement. Each time, we randomly select a record i in D with probability $\frac{1}{|D|}$, put i into the new dataset D' and then throw i back into D until $|D'| = |D|$. The probabilities that each record in D appears in D' are the same. So, from DP point of view, each DT in RF should be treated as being generated from the original data. So, we should provide DP protection on each DT in RF. In real-world industrial DM services, RF has shown its advantages on scalability, stability and good accuracy, which make RF widely used.[2] This motivates us to analyze and evaluate the DP techniques for DT based models.

4.3 DP for Decision Trees

We describe industrial DP implementations for DTs within three basic architectures.

4.3.1 Data Publication Architecture for DTs

The Data Publication Architecture publishes a synthetic dataset with DP guarantee. Any DM algorithm including DT can directly run on the published data. The design concept of this architecture is shown in Figure 6. First, a schema which represents the original dataset D's structure is constructed under DP. *Exponential mechanism* is used to avoid the privacy leakage from the schema's structure. *Laplace mechanism* is used to generate the parameters within the schema. Second, a new synthetic dataset D' is generated from this schema and is published to the DM service. The detailed implementation description of this architecture can be found in [25].

4.3.2 Separated Architecture for DTs

In Algorithm 1, the key step of a DT construction is to find the proper feature to create new branches. Although different algorithms use different impurity estimation measurements, they share one common characteristic, which is the utilization of the counting

[2]In large scale industrial systems, Logistic Regression (LR) with deep feature engineering is widely used for high dimensional sparse data and RF is widely used for relatively Low dimensional dense data.

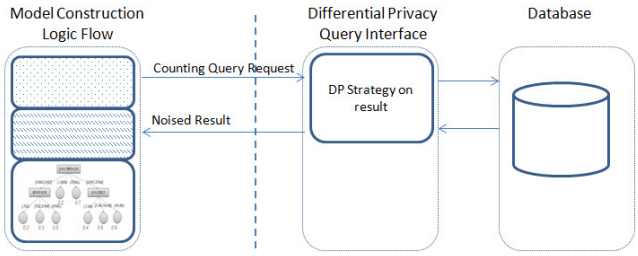


Figure 7: Separated Architecture for DTs.

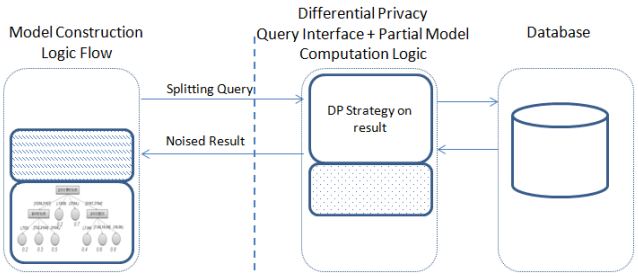


Figure 9: Hybridized Architecture for DTs.

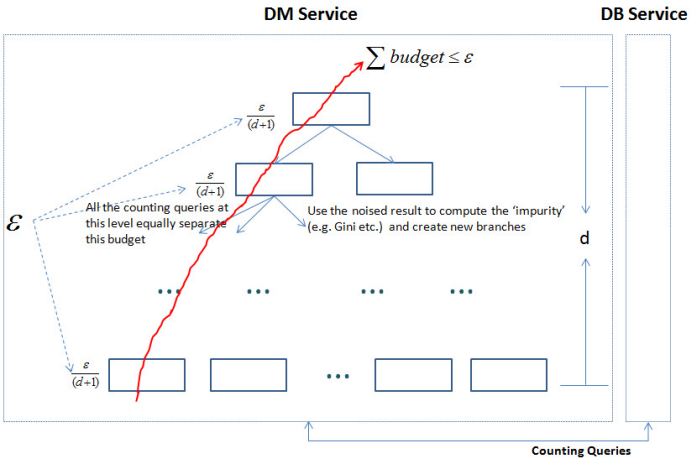


Figure 8: Implementation of the Separated Architecture.

query results to implement the calculation. For example, to compute the entropy of a dataset D, the algorithm checks D to obtain instances with labels $l \in L$. So, if D provides a DP interface to support counting quires, a DT can be constructed from the noisy counting results.

The design concept of the Separated Architecture implementation for DTs is shown in Figure 7, where the DB service does not have knowledge about the DM service. It only provides a DP interface for counting queries. The DM service uses the noise counting query results to compute the "impurity" measurements and then selects the proper feature to create new branches. Figure 8 demonstrates a detailed implementation of this solution. The privacy budget ϵ is equally divided for each layer of nodes. A node would equally divide its privacy budget to the counting queries issued by itself. Since there is no overlap on the datasets queried by the nodes in different branches, the DP solution only needs to control the sum of privacy budget consumed by each "path" (aka a path from root node to leaf node) is at most ϵ. Each node gets a privacy budget $\epsilon/(d+1)$, where d is the pre-defined maximum depth of the DT. When the generation of DT reaches the maximum depth, the computation must stop to satisfy the DP requirement. Here we only demonstrate the basic design idea. There are many details about how to assign the privacy budget to quires as well as tree prunes. In this architecture, the DB service only needs to use the *Laplace Mechanism* to support counting queries under DP. Compared to the Data Publication Architecture, the DT's construction directly uses the counting queries. So, the accuracy of the constructed DT is expected to be better. However, the DM service is limited by the query types provided by DB, while the Data Publication Architecture can support any DM services.

4.3.3 *Hybridized Architecture for DTs*

We follow the design of paper [7] to implement this architecture. When building a DT, the results of counting queries are used to represent the impurity of splitting node, which are used to guide the selection of feature to create new branches. So, the impurity computation is an intermediate step. The database can directly answer a query, through which we could select the best feature to split the training records (for continuous value features, also include the splitting node, such as creating two branches with ranges $(0, 3]$ and $(3, 100)$). In this case, the current node and the impurity measurement are the input of the query, and the output is a feature (including the splitting node for continuous value features). The above solution matches exactly the using scenario of the *Exponential Mechanism* in DP, where the quality function of the query is the impurity measurement. The exponential mechanism samples a branching solution according to the probability distribution based on the impurity quality function. The design concept of the Hybridized Architecture is shown in Figure 9.

A detailed demonstration is illustrated in Figure 10. Similar to the Separated Architecture, the privacy budget ϵ is equally divided for each layer of nodes. An inner node will use this privacy budget to directly ask DB to provide the best splitting selection. Interested readers can find the detailed information in paper [7]. Compared with the Separated Architecture, the influences of the noises are expected to be smaller in the Hybridized Architecture. The Separated Architecture uses the noise counting query results to compute the impurity measures to find the splitting node. while the impurity measure calculation of the Hybridized Architecture is accurate. After the impurity measures are computed, the noise is added through the exponential mechanism. So, the noise influences are expected to be smaller. The shortcoming of this solution is that the DB provider must also handle some DM calculations, such as the impurity measurement (entropy or Gini index) calculation. Therefore, we need to develop customized DB to support different DM services. This reduces the versatility of DB and brings DM complexity to DB.

4.4 Computational Complexity

Privacy protection techniques bring extra burden to both DB and DM services. For the Data Publication Architecture, the DM will be operated on the synthetic dataset, so there is no extra running costs except the one time cost to generate the synthetic data. For the Separated Architecture, the DB service needs to add Laplace noise for every counting query, so it at most doubles the computational complexity of the original DT construction. The Hybridized Architecture computes the quality of each possible splitting node and uses the exponential mechanism to select one. The only extra computational cost is the exponential mechanism selection process when compared with the Separated Architecture. So, it at most

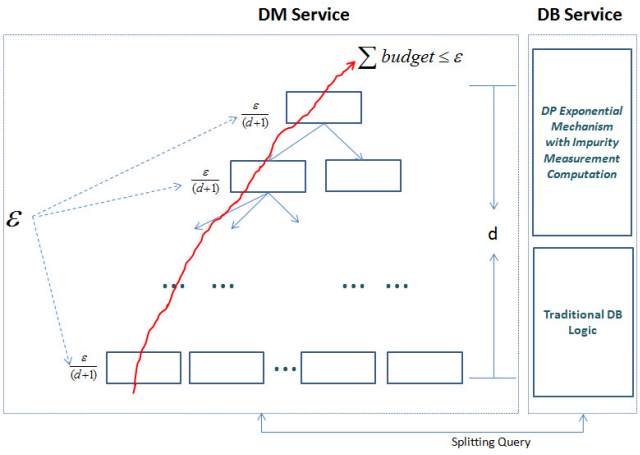


Figure 10: Implementation of the Hybridized Architecture.

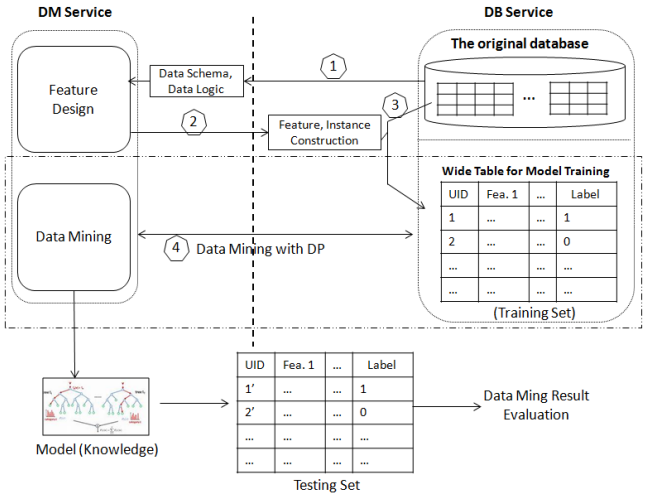


Figure 11: The evaluation procedure of different DP architectures.

triples the computational complexity. However, it should be emphasized that the DM services such as churn prediction do not have real-time requirement. Monthly classifier updating is enough to support the business value [12]. As a result, the computational complexity of DP is not an important issue.

5. EXPERIMENTAL RESULTS

Figure 11 shows how to evaluate different DP techniques in the deployed churn prediction system. First, the DB service publishes data schema and data logic introductions to the DM service, which in turn designs and sends the feature extraction scripts to the DB. Second, the DB service generates a unified customer-feature wide table, which consists of a customer ID, a group of feature vectors $[f_1, f_2, \cdots, f_m]$, and a class label $l = \{0, 1\}$ ($l = 1$ for churner). Different DP solutions are applied to protect each individual customer's privacy in this table. All DM algorithms run on the protected customer-feature wide table for both training and testing. After a prediction model is learned from training data, there is a testing set to evaluate its prediction performance under a certain privacy guarantee. We use the prediction performance without privacy protection as the baseline to evaluate different DP solutions.

Table 3: Statistics of churners (9 months from 2013-12).

	Month 1	Month 2	Month 3	Month 4	Month 5
Churner	185779	173576	196984	184728	216010
No-Churner	1927748	1935496	1907548	1909698	1893469
Total	2113527	2109072	2104532	2094426	2109479
	Month 6	Month 7	Month 8	Month 9	
Churner	201374	200492	199456	202873	
No-Churner	1909472	1918349	1983917	1949832	
Total	2110846	2118841	2183373	2152705	

For simplicity, we do not perform recursive feature engineering and build prediction model once on the protected customer-feature table in experiments. However, in real-world industrial systems, there are several rounds of feature engineering and the model will be refined several times to reach the best one. In this situation, the privacy budget should be divided to each round. Another point we should emphasize about the evaluation system is that our reporting is based on one month prediction system. If the process is repeated for every month and the time windows for training are overlapped, a customer will probably be present in multiple months, thus each month's task in not independent. Therefore, the privacy budget should be divided among the months.

Table 3 shows the basic statistics of experimental dataset, which is collected from one of biggest telco operators in China, having 9 consecutive months of more than 2 million prepaid customer's behavior records from 2013 to 2014. In each month, the number of churners takes around 9.2% of the total number of customers in the dataset. As shown in Figure 3, such an experiment can repeat in the sliding window and the average prediction results are reported. For each prediction task, we also repeat our experiment 5 times and report the average performance to avoid the influence of randomness.

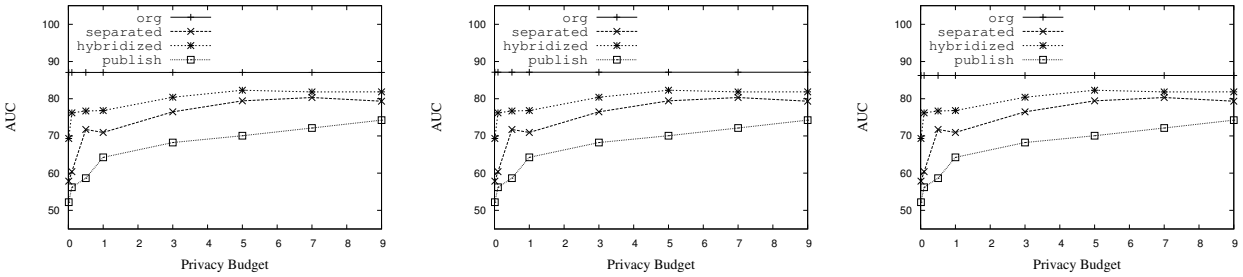
To evaluate the prediction performance, we use the area under the ROC curve (AUC) [9, 12] on the testing dataset, which is the standard performance metric in most prediction systems. The AUC is calculated as follows,

$$\text{AUC} = \frac{\sum_{n \in \text{true churners}} Rank_n - \frac{P \times (P+1)}{2}}{P \times N}, \qquad (1)$$

where P is the number true churners and N the number of true non-churners. Sorting the churner likelihood in descending order, we assign the highest likelihood customer with the rank n, the second highest likelihood customer with the rank $n-1$, and so on. The higher AUC indicates the better prediction performance.

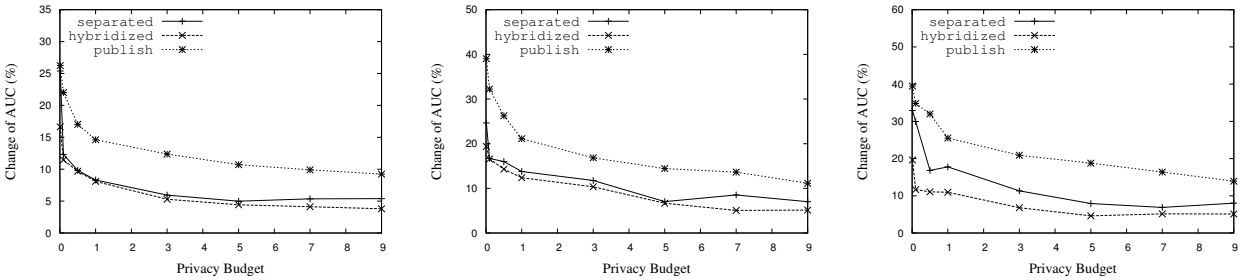
We aim to evaluate three basic DP architectures on the DT algorithms [7, 25] in terms of the overall predictive performance. The more complex data mining algorithm, such as RF [2], is composed of a group of DTs. So, it is essential to test how a single DT performs under three basic DP architectures. More specifically, we examine the following aspects of DP:

- To evaluate how the DP solutions perform with different privacy guarantees, we change the privacy budget parameter ϵ from 0.01 to 9.0 on 1 million training instances to examine the AUC's change of a DT.
- To study the DP solutions' performance on the variety of features in training data, we test a series of DTs trained from top 5 features to top 70 features.
- To examine the sensitivity of DP solutions to the training data volume, we train DTs with the data volume (number of train-



(a) Top 8 Features (b) Top 16 Features (c) Top 30 Features

Figure 12: Comparisons for different privacy budget ϵ with 1 million training instances (AUC).



(a) Top 8 Features (b) Top 16 Features (c) Top 30 Features

Figure 13: Comparisons for different privacy budget ϵ with 1 million training instances (relative AUC loss).

ing instances) from 0.01M to 2M with different privacy budget parameters.

- After analyzing results of the single DT, we report experimental results on the DP with RF as well.

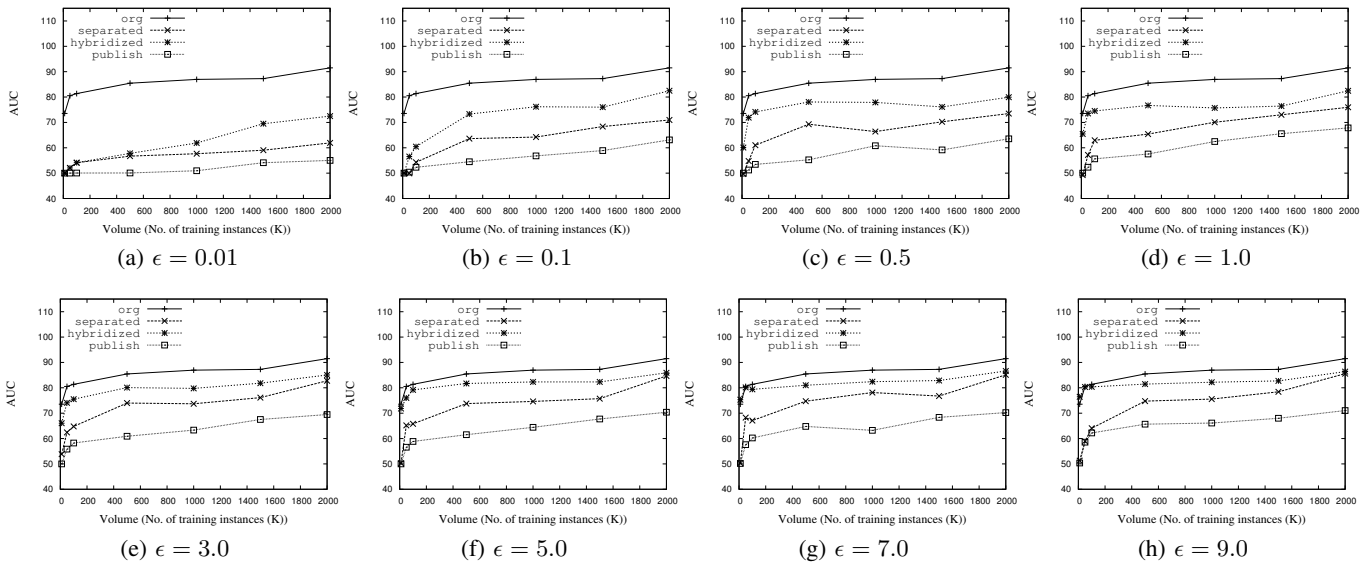
5.1 Privacy Budget

Figure 12 shows the AUCs when using 8, 16 and 30 top important features to construct DTs, respectively. The top features are ranked by accumulating impurity values of all nodes in DTs for each feature. In this experiment, we use 1 million instances to train the DT. We change the privacy budget ϵ from 0.01 to 9. Obviously with the increasing of privacy budget, the prediction accuracy loss (decrease of AUC) becomes smaller. The Hybridized Architecture outperforms the Separated Architecture, and the Separated Architecture outperforms the Data Publication Architecture. The relative percentage of AUC drop compared to that of no privacy protection is shown in Figure 13. From the results, we see that with the increase of privacy budget parameter, AUC loss quickly reduces because less privacy will be protected. We also find that with the growth of feature variety (the number of top features used to generate the DT), the AUC loss becomes larger.

As shown in Figure 13, we see that for our DM service with 1 million training instances, the performance of the privacy preserving algorithms can be very close (less than 5% AUC loss) to the original system without privacy protection when the privacy budget is above 3. However, when selecting $\epsilon \geq 3$, it only guarantees $\frac{Pr[A(D_1) \in O]}{Pr[A(D_2) \in O]} \leq e^{\epsilon} \geq 20$. In this way, DP may not work well because the two probabilities may have significant difference (distinguishable), and the adding/removing of an individual record may be very likely to be detected. If we set the small privacy budget parameter 0.1 or 0.01 as recommended by [7], the relative AUC loss is as large as $15\% \sim 30\%$. This is usually an unacceptable accuracy loss in real-world industrial DM services. Note that the DM service in our experimental settings is still an ideal case, in which only one round DM is performed on the customer-feature table. In most of the complex practical applications, the customer-feature wide table cannot be generated in one round and the recursive feature engineering is often needed. As a result, several rounds of DM services are required, which causes much smaller privacy budget parameter for each round when compared with one round evaluation in this paper. This implies that we still need lots of explorations to realize the practical deployment of DP techniques.

5.2 Big Data: Volume and Variety

We also test the DP algorithms using different volume of training data. The results are shown in Figure 14. We change the number of training instances from 0.01 million to 2 million. We observe that with the increasing volume of training data, all the architectures perform continuously better and the Hybridized Architecture performs the best. With the increase of privacy budget parameters, the performances of different architectures are becoming closer. From these results, we find that the AUC loss of privacy protection decreases with the increase of data volume. We explain this phenomenon from the DP's design concept. Since the amount of noise to be added is determined by the query/function characteristics (the global sensitivity of the specific query/function), the noises added for different volume of training data are in the same range. However, the absolute value of a query/function's output becomes larger in a large volume of data than the small volume of data. The small volume of data are more sensitive to the same amount of noise than the large volume of data. So, with the growth of training data volume, the AUC loss decreases. Since the noise to guarantee DP is created according to a distribution, the very small privacy budget parameter means the noise selection range becomes quite large,



(a) $\epsilon = 0.01$ (b) $\epsilon = 0.1$ (c) $\epsilon = 0.5$ (d) $\epsilon = 1.0$

(e) $\epsilon = 3.0$ (f) $\epsilon = 5.0$ (g) $\epsilon = 7.0$ (h) $\epsilon = 9.0$

Figure 14: Comparisons for different data volume (no. of training instances).

which also leads to small amount of random variations. We see that there indeed exist some random variations when the privacy budget parameter is too small (e.g., $\epsilon = 0.01$).

Figure 15 shows the experimental results with different privacy budget parameters and the number of top features. We can observe that with the increase of features, the performance of DP architectures decreases. This is because the model complexity increases with the variety growth of features. When there are more features, more queries are required to construct the model. Thus, the privacy budget assigned for each query becomes smaller, which causes the performance of the DM service drops due to more introduced noise. If we increase variety of features by using from the top 5 to top 70, the prediction performance without privacy protection increases slightly without much variation. The reason of this phenomenon is that a small number of top features determine the DT structures so as to dominate the overall prediction performance.

5.3 Random Forest v.s. Decision Trees

We present some evaluation results between RF and DT. Figure 16 shows the results of RFs with 10 DTs in three DP architectures. We see that if no privacy protection is required, the RF performs much better than a single DT. However, after adding privacy protection, the performance of RF becomes worse than a single DT. This is because we need to divide the privacy budget to each DT in the RF, and each DT can only use a very small privacy budget. In our experiment, each DT in RF only uses $1/10$ privacy budget, which makes the RF with privacy protection perform worse. This result inspires us to design more specific DP implementations for RF, which will be discussed in the next section.

6. DISCUSSIONS AND CONCLUSIONS

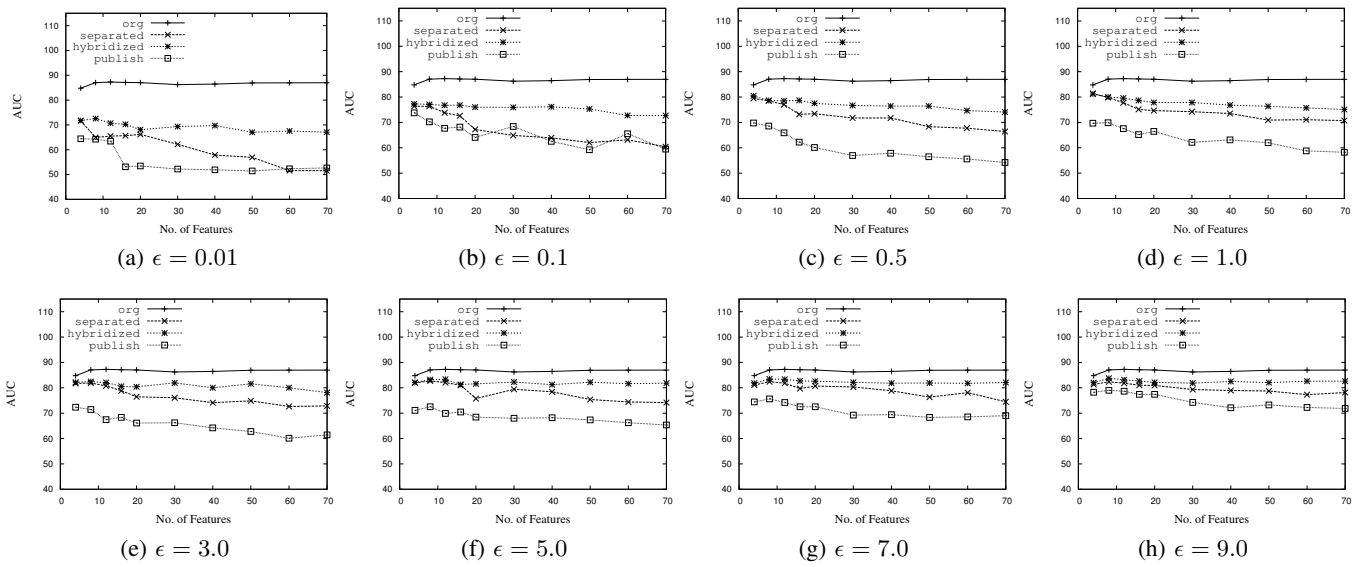
In this work, we implement and evaluate DP with three different architectures in a real deployed large-scale telco industrial system. We demonstrate some important observations from the extensive experimental results and give some suggestions for future work. We hope this work can help the possible practical deployment of the privacy protection solutions in the future.

From the previous experiments, we can find that, with one exception when $\epsilon = 0.01$, the Hybridized Architecture performs better than the Separated Architecture, while the Separated Architecture performs better than the Data Publication Architecture. This result is consistent with the analysis of different architecture design concepts. We can conclude that for the effect of data mining, the Hybridized Architecture performs better than the Separated Architecture and the Separated Architecture performs better than the Data Publication Architecture, while, from the system flexibility point of view, the Data Publication Architecture is better than the Separated Architecture and the Separated Architecture is better than the Hybridized Architecture.

From the analysis and evaluation, we can summarize three possible future directions to make DP practically usable in large-scale industrial systems.

- Relaxing the privacy guarantee (e.g., increasing privacy budget parameter ϵ) and studying its effectiveness on specific industrial applications. It is interesting to check what privacy leakage is tolerable in real industrial systems.
- Designing a specific privacy scheme for a certain data mining algorithm.
  - The Hybridized Architecture can possibly be improved by adapting the DP on the entire DM, which may be implemented in the following algorithm with two steps:
    - ◇ Using the Exponential Mechanism to select a tree structure. Each tree has a quality function to evaluate how impurity it divides the space. The challenge here is that there exists a large number of possible trees. Generating all of them is not computational possible. One possible solution is making use of the Markov Decision Process [34] to find a proper tree by only sampling part of the tree space.
    - ◇ Making use of the Laplace Mechanism to assign proper probability values in the final published tree (On the leaf nodes).

    We note that the privacy budget can be divided into two parts to serve these two steps respectively.



(a) $\epsilon = 0.01$ (b) $\epsilon = 0.1$ (c) $\epsilon = 0.5$ (d) $\epsilon = 1.0$

(e) $\epsilon = 3.0$ (f) $\epsilon = 5.0$ (g) $\epsilon = 7.0$ (h) $\epsilon = 9.0$

Figure 15: Comparisons for different number of features.

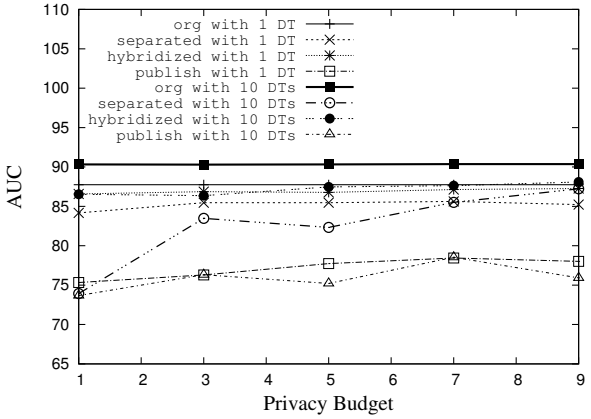


Figure 16: Experiments on random forest.

– Adaptively dividing the privacy budget. The current implementations equally divide the privacy budget to each layer of the DT. However, if a node is near to the root node, there may exist much more instances than a node near to leaves. Thus a node which is nearer to the leaves is more sensitive to the noise. Comparing to the current solution, an algorithm which assigns smaller privacy budget to the nodes near to the root node than the nodes near to the leaves, would perform better. So, a possible improvement direction of DP algorithms is to design adjustable privacy budget assignment strategies.

– Designing the tradeoff DP Mechanism for RF model. If we divide the dataset and build DTs on subsets of the original dataset to implement the RF, the performance of the model may be decreased. However, by separating the dataset into on-overlapping parts, we can assign ϵ to each part to build RF. Thus each DT will get a larger privacy budget. A tradeoff DP Mechanism can be designed for RF.

- Using large volume of data but with low variety for model training. It is an interesting problem to check how large is enough to take the benefit of reducing accuracy loss by increasing data volume.

Acknowledgments

This work was supported in part by National Grant Fundamental Research (973 Program) of China under Grant 2014CB340304 and 2012-CB316200, Microsoft Research Asia Gift Grant and Google Faculty Award 2013, Shanghai Municipal Science and Technology Commission Project 14510722600, NSFC (Grant No. 61373092, 61033013 and 61328202), Hong Kong RGC project 620812 and N_HKUST63713, and Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 12KJA520004). This work was partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization.

7. REFERENCES

[1] K. binti Oseman, S. binti Mohd Shukor, N. A. Haris, and F. bin Abu Bakar. Data mining in churn analysis model for telecommunication industry. *Journal of Statistical Modeling and Analytics*, 1:19–27, 2010.

[2] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[3] L. H. Cox. Suppression methodology and statistical disclosure control. *Journal of the American Statistical Association*, 75(370):377–385, 1980.

[4] T. Dalenius. nding a needle in a haystack or identifying anonymous census records. *Journal of Official Statistics*, 2(3):329–336, 1986.

[5] J. Domingo-Ferrer and V. Torra. A Critique of k-Anonymity and Some of Its Enhancements. In *Third International Conference on Availability, Reliability and Security (ARES 08)*, pages 990–993. IEEE, 2008.

[6] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

[7] A. Friedman and A. Schuster. Data mining with differential privacy. In *Proceedings of the 16th ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining, KDD '10, pages 493–502, New York, NY, USA, 2010. ACM.

[8] A. Friedman and A. Schuster. Data mining with differential privacy. In B. Rao, B. Krishnapuram, A. Tomkins, and Q. Yang, editors, *KDD*, pages 493–502. ACM, 2010.

[9] I. Guyon, V. Lemaire, M. Boullé, G. Dror, and D. Vogel. Analysis of the KDD Cup 2009: Fast scoring on a large orange customer database. 7:1–22, 2009.

[10] J. Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[11] C. S. Hien To, Gabriel Ghinita. A framework for protecting worker location privacy in spatial crowdsourcing. *Proc. VLDB Endow.*, 10(7):919–930, 2014.

[12] Y. Huang, F. Zhu, M. Yuan, K. Deng, Y. Li, B. Ni, W. Dai, Q. Yang, and J. Zeng. Telco churn prediction with big data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 607–618, Melbourne, VC, AUS, 2015. ACM.

[13] G. Jagannathan, K. Pillaipakkamnatt, and R. N. Wright. A practical differentially private random decision tree classifier. *Trans. Data Privacy*, 5(1):273–295, Apr. 2012.

[14] S. Jiang, G. A. Fiore, Y. Yang, J. Ferreira Jr, E. Frazzoli, and M. C. González. A review of urban computing for mobile phone traces: current methods, challenges and opportunities. In *KDD Workshop on Urban Computing*, pages 2–9, 2013.

[15] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, SIGMOD '05, pages 49–60, New York, NY, USA, 2005. ACM.

[16] A. Lemmens and C. Croux. Bagging and boosting classification trees to predict churn. *Journal of Marketing Research*, 43(2):276–286, 2006.

[17] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, pages 106–115. IEEE, 2007.

[18] N. Li, W. Qardaji, D. Su, and J. Cao. Privbasis: Frequent itemset mining with differential privacy. *Proc. VLDB Endow.*, 5(11):1340–1351, July 2012.

[19] E. Lima, C. Mues, and B. Baesens. Domain knowledge integration in data mining using decision tables: Case studies in churn prediction. *Journal of the Operational Research Society*, 60(8):1096–1106, 2009.

[20] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD'08*, pages 93–106, 2008.

[21] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In L. Liu, A. Reuter, K.-Y. Whang, and J. Zhang, editors, *ICDE*, page 24. IEEE Computer Society, 2006.

[22] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[23] A. Machanavajjhala, A. Korolova, and A. D. Sarma. Personalized social recommendations: Accurate or private. *Proc. VLDB Endow.*, 4(7):440–450, Apr. 2011.

[24] F. D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, SIGMOD '09, pages 19–30, New York, NY, USA, 2009. ACM.

[25] N. Mohammed, R. Chen, B. C. Fung, and P. S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 493–501, New York, NY, USA, 2011. ACM.

[26] C. Phua, H. Cao, J. B. Gomes, and M. N. Nguyen. Predicting near-future churners and win-backs in the telecommunications industry. *arXiv preprint arXiv:1210.6891*, 2012.

[27] W. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, 0:757–768, 2013.

[28] S. Rendle. Scaling factorization machines to relational data. In *PVLDB*, volume 6, pages 337–348, 2013.

[29] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):571–588, Oct. 2002.

[30] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[31] E. Ulfar, P. Vasyl, and K. Aleksandra. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *21st ACM Conference on Computer and Communications Security*, pages 1054–1067, 2014.

[32] C.-P. Wei and I. Chiu. Turning telecommunications call details to churn prediction: a data mining approach. *Expert systems with applications*, 23(2):103–112, 2002.

[33] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 543–554. VLDB Endowment, 2007.

[34] Q. Xiao, R. Chen, and K.-L. Tan. Differentially private network data release via structural inference. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 911–920, New York, NY, USA, 2014. ACM.

[35] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, ICDE '12, pages 32–43, Washington, DC, USA, 2012. IEEE Computer Society.

[36] H.-F. Yu, H.-Y. Lo, H.-P. Hsieh, J.-K. Lou, T. G. McKenzie, J.-W. Chou, P.-H. Chung, C.-H. Ho, C.-F. Chang, Y.-H. Wei, et al. Feature engineering and classifier ensemble for kdd cup 2010. In *JMLR W & CP*, pages 1–16, 2010.

[37] M. Yuan, L. Chen, and P. S. Yu. Personalized privacy protection in social networks. *Proc. VLDB Endow.*, 4:141–150, November 2010.

[38] C. Zeng, J. F. Naughton, and J.-Y. Cai. On differentially private frequent itemset mining. *Proc. VLDB Endow.*, 6(1):25–36, Nov. 2012.

[39] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: Private data release via bayesian networks. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 1423–1434, New York, NY, USA, 2014. ACM.